Amendments to the Claims

Claims 10, 18 and 29 have been amended. The remaining claims remain unchanged.

1-9. (Canceled)

10. (Currently Amended) In a Java™ computing environment, a method of identifying active Java™ objects and active Java™ classes by a virtual machine at runtime during garbage collection, said method comprising:

generating and loading in the virtual machine prior to execution time a cluster of Java™ object representations which are sequentially represented inside the virtual machine, wherein each of said Java™ object representations in said cluster consists of:

a first reference to an internal class representation of a class associated with a Java™ object, and

a second reference to instance fields associated with said Java™ object;

sequentially reading by said virtual machine at runtime said cluster of Java™ object representations;

determining by said virtual machine at runtime during garbage collection whether Java™ objects or Java™ classes are to be identified;

using said second references of said cluster during garbage collection to mark memory addresses that correspond to Java™ objects when said determining determines that Java™ objects are to be identified, thereby allowing Java™ objects to be identified at ~~run time~~ runtime by a sequential read of said cluster; and

using one or more of said first references of said cluster during garbage collection to mark memory addresses that correspond to Java™ classes when said determining determines that Java™ classes are to be identified, thereby allowing Java™ classes to be identified at ~~run time~~ runtime during garbage collection by a sequential read of said cluster, wherein said first reference is a direct reference to said internal class representation of said Java™ object.

11. (Previously Presented) A method as recited in claim 10,

wherein said second reference is a reference to an array of references, and

wherein each reference in said array of references is a reference to an instance field associated with said Java™ object.

12. (Previously Presented) A method as recited in claim 10, wherein said first and second references are allocated as four bytes.

13. (Previously Presented) A method as recited in claim 10, wherein said method further comprises:

removing internal class representations that have not been marked.

14. (Previously Presented) A method as recited in claim 10, wherein said method further comprises:

removing Java™ objects that have not been marked.

15-17. (Canceled)

18. (Currently Amended) A computer readable medium including at least computer program code for identifying active Java™ objects and active Java™ classes by a virtual machine at runtime during garbage collection, comprising:

computer program code for generating and loading in the virtual machine prior to execution time a cluster of Java™ object representations which are sequentially represented inside the virtual machine, wherein each of said Java™ object representations in said cluster consists of:

a first reference to an internal class representation of a class associated with a Java™ object, and

a second reference to instance fields associated with said Java™ object;

computer program code for sequentially reading by said virtual machine at runtime during garbage collection said cluster of Java™ object representations;

computer program code for determining by said virtual machine at runtime during garbage collection whether Java™ objects or Java™ classes are to be identified;

computer program code for using said second references of said cluster during garbage collection to mark memory addresses that correspond to Java™ objects when said determining determines that Java™ objects are to be identified, thereby allowing

Java™ objects to be identified at ~~run-time~~ runtime by a sequential read of said cluster; and

computer program code for using one or more of said first references of said cluster during garbage collection to mark memory addresses that correspond to Java™ classes when said determining determines that Java™ classes are to be identified, thereby allowing Java™ classes to be identified at ~~run-time~~ runtime during garbage collection by a sequential read of said cluster, wherein said first reference is a direct reference to said internal class representation of said Java™ object.

19. (Previously Presented) A computer readable medium as recited in claim 18,

wherein said second reference is a reference to an array of references, and

wherein each reference in said array of references is a reference to an instance field associated with said Java™ object.

20. (Previously Presented) A computer readable medium as recited in claim 19, wherein said first and second references are allocated as four bytes.

21-24. (Canceled)

25. (Previously Presented) A computer-readable medium as recited in claim 18, further comprising:

computer program code for removing internal class representations that have not been marked.

26. (Previously Presented) A computer-readable medium as recited in claim 18, further comprising:

computer program code for removing Java™ objects that have not been marked.

27. (Previously Presented) A computer-readable medium as recited in claim 18, wherein said Java™ objects are identified for garbage collection at runtime.

28. (Canceled)

29. (Currently Amended) In a Java™ computing environment, a computer system for identifying active Java™ objects and active Java™ classes by a virtual machine at runtime during garbage collection, said computer system comprising:

memory;

at least one processor which is configured to:

generate and load in the virtual machine prior to execution time a cluster of Java™ object representations which are sequentially represented inside the virtual machine, wherein each of said Java™ object representations in said cluster consists of: a first reference to an internal class representation of a class associated with a Java™ object, and a second reference to instance fields associated with said Java™ object;

sequentially reading by said virtual machine at runtime said cluster of Java™ object representations;

~~determinie~~ determine by said virtual machine at runtime during garbage collection whether Java™ objects or Java™ classes are to be identified;

use ~~using~~ said second references of said cluster during garbage collection to mark memory addresses that correspond to Java™ objects when said determining determines that Java™ objects are to be identified, thereby allowing Java™ objects to be identified at ~~run-time~~ runtime by a sequential read of said cluster; and

use one or more of said first references of said cluster during garbage collection to mark memory addresses that correspond to Java™ classes when said determining determines that Java™ classes are to be identified, thereby allowing Java™ classes to be identified at ~~run-time~~ runtime during garbage collection by a sequential read of said cluster, wherein said first reference is a direct reference to said internal class representation of said Java™ object.

30. (Previously Presented) A computer system as recited in claim 29,

wherein said second reference is a reference to an array of references, and

wherein each reference in said array of references is a reference to an instance field associated with said Java™ object.

31. (Previously Presented) A computer system as recited in claim 29, wherein said first and second references are allocated as four bytes.

32. (Previously Presented) A computer system as recited in claim 29, wherein said at least one processor is further configured to remove internal class representations that have not been marked.

33. (Previously Presented) A computer system as recited in claim 29, wherein said at least one processor is further configured to remove Java™ objects that have not been marked.